Rewrite Semantics for Guarded Recursion

Patrick Bahr

IT University of Copenhagen

joint work with Rasmus Møgelberg

Overview

Guarded Recursive Types

Dependent Types

Reduction Semantics

Guarded Recursive Types

H. Nakano. A modality for recursion. In: LICS, 2000

Guarded Recursion

- type modality $\triangleright$ (pronounced "later")
- $\triangleright$ is an applicative functor

$$\text{next} : A \to \triangleright A$$

$$\circledast : \triangleright(A \to B) \to \triangleright A \to \triangleright B$$

- fixed-point operator $\text{fix} : (\triangleright A \to A) \to A$
- guarded recursive types: $\mu X.A$

Example

guarded streams

$$\mathsf{Str} = \mu X.\mathsf{Nat} \times \triangleright X$$

$$\mathsf{cons} : \mathsf{Nat} \to \triangleright\mathsf{Str} \to \mathsf{Str}$$

$$\mathsf{cons} = \lambda x.\lambda y.\langle x, y\rangle$$

$$\mathsf{ones} : \mathsf{Str}$$

$$\mathsf{ones} = \mathsf{fix}(\lambda x.\mathsf{cons}\,1\,x)$$

Motivation

- functional reactive programming
- productive coprogramming
  (clocks & clock quantification)
- solving recursive domain equations
  ($\rightarrow$ synthetic domain theory)

Motivation

- functional reactive programming
- productive coprogramming
  (clocks & clock quantification)
- solving recursive domain equations
  ($\rightarrow$ synthetic domain theory)

$$X \simeq \mathsf{Nat} + (\quad X \times \quad X) + (\quad X \rightarrow \quad X)$$

Motivation

- functional reactive programming
- productive coprogramming
  (clocks & clock quantification)
- solving recursive domain equations
  ($\rightarrow$ synthetic domain theory)

$$X \simeq \mathsf{Nat} + (\triangleright X \times \triangleright X) + (\triangleright X \rightarrow \triangleright X)$$

Dependent Types

A. Bizjak, H. B. Grathwohl, R. Clouston, R. E. Møgelberg, and L. Birkedal. Guarded dependent type theory with coinductive types. In FoSSaCS, 2016.

Combining Π and $\triangleright$

$$\frac{\Gamma \vdash s : \Pi x : A.B \qquad \Gamma \vdash t : A}{\Gamma \vdash s\, t : B\,[t/x]}$$

Combining Π and $\triangleright$

$$\frac{\Gamma \vdash s : \Pi x : A.B \qquad \Gamma \vdash t : A}{\Gamma \vdash s\, t : B\,[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : ???}$$

Combining Π and $\triangleright$

$$\frac{\Gamma \vdash s : \Pi x : A.B \qquad \Gamma \vdash t : A}{\Gamma \vdash s\, t : B\,[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\, B\,[t/x]}$$

Combining Π and $\triangleright$

$$\frac{\Gamma \vdash s : \Pi x : A.B \qquad \Gamma \vdash t : A}{\Gamma \vdash s\, t : B\,[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\, B\,[t/x]}$$

- Problem: $t : \triangleright\, A$, but $x : A$

Combining Π and $\triangleright$

$$\frac{\Gamma \vdash s : \Pi x : A.B \qquad \Gamma \vdash t : A}{\Gamma \vdash s\, t : B\,[t/x]}$$

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\, B\,[t/x]}$$

- Problem: $t : \triangleright\, A, \quad$ but $x : A$
- needed: getting rid of $\triangleright$ in a controlled way

Delayed Substitutions

Instead of

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\, B\,[t/x]}$$

we have

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\,[x \leftarrow t]\,.B}$$

Delayed Substitutions

Instead of

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\, B\,[t/x]}$$

we have

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\,[x \leftarrow t]\,.B}$$

In general

$$\triangleright\,[x_1 \leftarrow t_1, \ldots x_n \leftarrow t_n]\,.A$$

Delayed Substitutions

Instead of

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\, B\,[t/x]}$$

we have

$$\frac{\Gamma \vdash s : \triangleright\,(\Pi x : A.B) \qquad \Gamma \vdash t : \triangleright\, A}{\Gamma \vdash s \circledast t : \triangleright\,[x \leftarrow t]\,.B}$$

In general

$$\triangleright\,[x_1 \leftarrow t_1, \ldots x_n \leftarrow t_n]\,.A$$

$$\mathsf{next}\,[x_1 \leftarrow t_1, \ldots x_n \leftarrow t_n]\,.t$$

Equalities

$$\triangleright\ [x \leftarrow \mathsf{next}\quad u].A = \triangleright\quad A[u/x]$$

Equalities

$$\triangleright\xi\,[x \leftarrow \mathsf{next}\xi.u]\,.A = \triangleright\xi.A\,[u/x]$$

Equalities

$$
\begin{aligned}
\triangleright\xi\,[x \leftarrow \mathsf{next}\xi.u]\,.A &= \triangleright\xi.A\,[u/x] \\
\triangleright\,\xi\,[x \leftarrow u]\,.A &= \triangleright\xi.A && \text{if } x \notin \mathsf{fv}(A) \\
\triangleright\,\xi\,[x \leftarrow u, y \leftarrow v]\,\xi'.A &= \triangleright\xi\,[y \leftarrow v, x \leftarrow u]\,\xi'.A && \text{if } \ldots
\end{aligned}
$$

Equalities

$$\begin{aligned}
\triangleright\xi\,[x \leftarrow \mathsf{next}\xi.u]\,.A &= \triangleright\xi.A\,[u/x] \\
\triangleright\,\xi\,[x \leftarrow u]\,.A &= \triangleright\xi.A && \text{if } x \notin \mathsf{fv}(A) \\
\triangleright\,\xi\,[x \leftarrow u, y \leftarrow v]\,\xi'.A &= \triangleright\xi\,[y \leftarrow v, x \leftarrow u]\,\xi'.A && \text{if } \ldots
\end{aligned}$$

$$\begin{aligned}
\mathsf{next}\xi\,[x \leftarrow \mathsf{next}\xi.u]\,.t &= \mathsf{next}\xi.t\,[u/x] \\
\mathsf{next}\xi\,[x \leftarrow u]\,.t &= \mathsf{next}\xi.t && \text{if } x \notin \mathsf{fv}(t) \\
\mathsf{next}\xi\,[x \leftarrow u, y \leftarrow v]\,\xi'.t &= \mathsf{next}\xi\,[y \leftarrow v, x \leftarrow u]\,\xi'.t && \text{if } \ldots
\end{aligned}$$

Typing rule

Simple Case

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : \triangleright A}{\Gamma \vdash \mathsf{next}\,[x \leftarrow u]\,.t : \triangleright\,[x \leftarrow u]\,.B}$$

Typing rule

Simple Case

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : \triangleright A}{\Gamma \vdash \mathsf{next}\,[x \leftarrow u]\,.t : \triangleright\,[x \leftarrow u]\,.B}$$

In General

$$\frac{\begin{array}{c}\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash t : B \\ \Gamma \vdash t_i : \triangleright\,[x_1 \leftarrow t_1, \ldots, x_{i-1} \leftarrow t_{i-1}]\,.A_i \text{ for all } 1 \leq i \leq n\end{array}}{\Gamma \vdash \mathsf{next}\,[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n]\,.t : \triangleright\,[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n]\,.B}$$

Applicative Structure

Applicative structure can be defined in terms of delayed substitutions:

$$s \circledast t = \text{next}\,[x \leftarrow s, y \leftarrow t]\,.x\,y$$

Applicative Structure

Applicative structure can be defined in terms of delayed substitutions:

$$s \circledast t = \text{next}\,[x \leftarrow s, y \leftarrow t]\,.x\,y$$

$$\begin{aligned}
&\text{next}\,u \circledast \text{next}\,v \\
=\ &\text{next}\,[x \leftarrow \text{next}\,u, y \leftarrow \text{next}\,v]\,.x\,y \\
=\ &\text{next}\,[x \leftarrow \text{next}\,u]\,.x\,v \\
=\ &\text{next}(u\,v)
\end{aligned}$$

Applicative Functor Laws

We need to add the following equality

$$\mathsf{next}\xi\,[x \leftarrow t]\,.x = t$$

Applicative Functor Laws

We need to add the following equality

$$\mathsf{next}\xi\,[x \leftarrow t]\,.x = t$$

We can then derive the applicative functor laws:

$$\begin{aligned}
\mathsf{next}(\lambda x.x) \circledast t &= t \\
\mathsf{next}(\lambda f.\lambda g.\lambda x.f\,(g\,x)) \circledast s \circledast t \circledast u &= s \circledast (t \circledast u) \\
\mathsf{next}\,s \circledast \mathsf{next}\,t &= \mathsf{next}\,(s\,t) \\
s \circledast \mathsf{next}\,t &= \mathsf{next}(\lambda f.f\,t) \circledast s
\end{aligned}$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma \vdash_{\Delta} t : B \qquad \kappa \in \Delta}{\Gamma \vdash_{\Delta} \mathsf{next}^{\kappa} t : \triangleright^{\kappa} B}$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash_\Delta t : B \qquad \kappa \in \Delta \qquad \Gamma \vdash_\Delta t_i : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_{i-1} \leftarrow t_{i-1}\right].A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_\Delta \text{next}^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].t : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].B}$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash_\Delta t : B \qquad \kappa \in \Delta \qquad \Gamma \vdash_\Delta t_i : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_{i-1} \leftarrow t_{i-1}\right].A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_\Delta \text{next}^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].t : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].B}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \Lambda\kappa.t : \forall\kappa.A} \qquad \frac{\Gamma \vdash_\Delta t : \forall\kappa.A \qquad \kappa' \in \Delta}{\Gamma \vdash_\Delta t[\kappa'] : A\left[\kappa'/\kappa\right]}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : \triangleright^\kappa A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \text{prev}\kappa.t : \forall\kappa.A}$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash_\Delta t : B \qquad \kappa \in \Delta \qquad \Gamma \vdash_\Delta t_i : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_{i-1} \leftarrow t_{i-1}\right].A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_\Delta \text{next}^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].t : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].B}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \Lambda\kappa.t : \forall\kappa.A} \qquad \frac{\Gamma \vdash_\Delta t : \forall\kappa.A \qquad \kappa' \in \Delta}{\Gamma \vdash_\Delta t[\kappa'] : A\left[\kappa'/\kappa\right]}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : \triangleright^\kappa A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \text{prev}\kappa.t : \forall\kappa.A}$$

$$\text{prev}\kappa.(\text{next}^\kappa \quad t) = \Lambda\kappa.t$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash_\Delta t : B \qquad \kappa \in \Delta \qquad \Gamma \vdash_\Delta t_i : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_{i-1} \leftarrow t_{i-1}\right].A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_\Delta \text{next}^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].t : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].B}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \Lambda\kappa.t : \forall\kappa.A} \qquad \frac{\Gamma \vdash_\Delta t : \forall\kappa.A \qquad \kappa' \in \Delta}{\Gamma \vdash_\Delta t[\kappa'] : A\left[\kappa'/\kappa\right]}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : \triangleright^\kappa A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \text{prev}\kappa.t : \forall\kappa.A}$$

$$\text{prev}\kappa.(\text{next}^\kappa \xi.t) = \Lambda\kappa.t(\text{adv}^\kappa\,(\xi))$$

Multiple Clocks and Clock Quantification

$$\frac{\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash_\Delta t : B \qquad \kappa \in \Delta \qquad \Gamma \vdash_\Delta t_i : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_{i-1} \leftarrow t_{i-1}\right].A_i \text{ for all } 1 \leq i \leq n}{\Gamma \vdash_\Delta \text{next}^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].t : \triangleright^\kappa \left[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\right].B}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \Lambda\kappa.t : \forall\kappa.A} \qquad \frac{\Gamma \vdash_\Delta t : \forall\kappa.A \qquad \kappa' \in \Delta}{\Gamma \vdash_\Delta t[\kappa'] : A\left[\kappa'/\kappa\right]}$$

$$\frac{\Gamma \vdash_{\Delta,\kappa} t : \triangleright^\kappa A \qquad \Gamma \vdash_\Delta}{\Gamma \vdash_\Delta \text{prev}\kappa.t : \forall\kappa.A}$$

$$\text{prev}\kappa.(\text{next}^\kappa \xi.t) = \Lambda\kappa.t(\text{adv}^\kappa\,(\xi))$$

$$\text{adv}^\kappa\,([x_1 \leftarrow t_1, \ldots x_n \leftarrow t_n]) = [(\text{prev}\kappa.t_1)[\kappa]/x_1, \ldots, (\text{prev}\kappa.t_n)[\kappa]/x_n]$$

Example

$$\mathsf{Str}^{\kappa} = \mu X.\mathsf{Nat} \times \triangleright^{\kappa} X$$
$$\mathsf{Str} = \forall \kappa.\mathsf{Str}^{\kappa}$$

$$\begin{aligned}
\mathit{tl} &: \mathsf{Str} \to \mathsf{Str} & \mathit{tl} &= \lambda x.\mathsf{prev}\kappa.\pi_2(x[\kappa]) \\
\mathit{hd} &: \mathsf{Str} \to \mathsf{Nat} & \mathit{hd} &= \lambda x.\pi_1(x[\kappa_0]) \\
\mathit{tln} &: \mathsf{Nat} \to \mathsf{Str} \to \mathsf{Str} & \mathit{tln} &= \lambda x.\lambda y.\mathsf{rec}\, x\, y\, (\lambda z_1.\lambda z_2.\mathit{tl}\, z_2) \\
\mathit{nth} &: \mathsf{Nat} \to \mathsf{Str} \to \mathsf{Nat} & \mathit{nth} &= \lambda x.\lambda y.\mathit{hd}\, (\mathit{tln}\, x\, y)
\end{aligned}$$

Example

$$\mathsf{Str}^{\kappa} = \mu X.\mathsf{Nat} \times \triangleright^{\kappa} X$$
$$\mathsf{Str} = \forall \kappa.\mathsf{Str}^{\kappa}$$

$$\begin{aligned} \mathit{tl} &: \mathsf{Str} \to \mathsf{Str} & \mathit{tl} &= \lambda x.\mathsf{prev}\kappa.\pi_2(x[\kappa]) \\ \mathit{hd} &: \mathsf{Str} \to \mathsf{Nat} & \mathit{hd} &= \lambda x.\pi_1(x[\kappa_0]) \\ \mathit{tln} &: \mathsf{Nat} \to \mathsf{Str} \to \mathsf{Str} & \mathit{tln} &= \lambda x.\lambda y.\mathsf{rec}\, x\, y\, (\lambda z_1.\lambda z_2.\mathit{tl}\, z_2) \\ \mathit{nth} &: \mathsf{Nat} \to \mathsf{Str} \to \mathsf{Nat} & \mathit{nth} &= \lambda x.\lambda y.\mathit{hd}\, (\mathit{tln}\, x\, y) \end{aligned}$$

$$\mathit{skip} : \mathsf{Str} \to \mathsf{Str}$$
$$\mathit{skip} = \lambda x.\Lambda\kappa.(\mathsf{fix}^{\kappa}(\lambda f.\lambda z.\, \langle \mathit{hd}\, z, \mathsf{next}^{\kappa}\, [y \leftarrow f]\, .y\, (\mathit{tl}\, (\mathit{tl}\, z))\rangle))\, x$$

Reduction Semantics

Motivation

- we want to implement a type checker for dependent type theory with guarded recursion
- we need to decide the equality theory
- possible approach: reduction relation that is
  - strongly normalising
  - confluent

Problems with Normalisation

- Fixed-point combinator!

$$\mathsf{fix}^{\kappa}\, t = t(\mathsf{next}^{\kappa}(\mathsf{fix}^{\kappa}\, t))$$

- We cannot turn this equation into a normalising rewrite rule:

$$\mathsf{next}\xi\, [x \leftarrow u, y \leftarrow v]\, \xi'.A = \mathsf{next}\xi\, [y \leftarrow v, x \leftarrow u]\, \xi'.A$$

Problems with Confluence

$$\mathsf{next}^{\kappa}\xi\,[x \leftarrow \mathsf{next}^{\kappa}\xi.s]\,.t = \mathsf{next}^{\kappa}\xi.t\,[s/x]$$

Problems with Confluence

$$\mathsf{next}^{\kappa}\xi\,[x \leftarrow \mathsf{next}^{\kappa}\xi.s]\,.t \rightarrow \mathsf{next}^{\kappa}\xi.t\,[s/x]$$

Problems with Confluence

$$\mathsf{next}^{\kappa}\xi\,[x \leftarrow \mathsf{next}^{\kappa}\xi.s]\,.t \rightarrow \mathsf{next}^{\kappa}\xi.t\,[s/x]$$

Problems with Confluence

$$\mathsf{next}^{\kappa}\xi\,[x \leftarrow \mathsf{next}^{\kappa}\xi.s]\,.t \rightarrow \mathsf{next}^{\kappa}\xi.t\,[s/x]$$

$$t = [x_1 \leftarrow y, x_2 \leftarrow [x_1 \leftarrow y]\,.0]\,.x_1x_2$$

$$\begin{array}{lcl} t & \rightarrow & \mathsf{next}^{\kappa}\,[x_1 \leftarrow y]\,.x_1 0 \\ t & \rightarrow & \mathsf{next}^{\kappa}\,[x_1 \leftarrow y, x_2 \leftarrow \mathsf{next}^{\kappa}.0]\,.x_1x_2 \end{array}$$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion freeze^{κ}: $\rhd^{\kappa} A \to A$.

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion $\mathsf{freeze}^{\kappa} : \triangleright^{\kappa} A \to A$.
- $\mathsf{next}^{\kappa}\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \mathsf{next}^{\kappa}\ u\,[\mathsf{freeze}^{\kappa}\,t/x]$
- $\triangleright^{\kappa}\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright^{\kappa}\ A\,[\mathsf{freeze}^{\kappa}\,t/x]$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion $\mathsf{freeze}^\kappa : \triangleright^\kappa A \to A$.
- $\mathsf{next}^\kappa\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \mathsf{next}^\kappa\, l.u\,[\mathsf{freeze}^\kappa_l\, t/x]$
- $\triangleright^\kappa\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright^\kappa l.A\,[\mathsf{freeze}^\kappa_l\, t/x]$

Alternative Calculus without Delayed Substitutions

Idea

- controlled conversion $\mathsf{freeze}^\kappa : \triangleright^\kappa A \to A$.
- $\mathsf{next}^\kappa\,[x \leftarrow t]\,.u \quad \rightsquigarrow \quad \mathsf{next}^\kappa\, l.u\,[\mathsf{freeze}^\kappa_l\, t/x]$
- $\triangleright^\kappa\,[x \leftarrow t]\,.A \quad \rightsquigarrow \quad \triangleright^\kappa\, l.A\,[\mathsf{freeze}^\kappa_l\, t/x]$

$$\frac{\Gamma \vdash^{\mathcal{L}}_{\Delta} t :_{\mathcal{I}} \triangleright^\kappa l.A \qquad l : \kappa \in \mathcal{L}}{\Gamma \vdash^{\mathcal{L}}_{\Delta} \mathsf{freeze}^\kappa_l\, t :_{\mathcal{I},l} A}$$

$$\frac{\Gamma \vdash^{\mathcal{L},l:\kappa}_{\Delta} t :_{\mathcal{I},l} A \qquad \kappa \in \Delta \qquad \Gamma \vdash_\Delta}{\Gamma \vdash^{\mathcal{L}}_{\Delta} \mathsf{next}^\kappa l.t :_{\mathcal{I}} \triangleright^\kappa l.A}$$

Other Typing Rules

$$\frac{\mathcal{J} \subseteq \mathcal{I}}{\Gamma, x :_{\mathcal{J}} A, \Gamma' \vdash^{\mathcal{L}}_{\Delta} x :_{\mathcal{I}} A} \qquad \frac{\Gamma, x :_{\mathcal{I}} A \vdash^{\mathcal{L}}_{\Delta} t :_{\mathcal{I}} B}{\Gamma \vdash^{\mathcal{L}}_{\Delta} \lambda x.t :_{\mathcal{I}} A \to B}$$

$$\frac{\Gamma \vdash^{\mathcal{L}}_{\Delta,\kappa} t :_{\mathcal{I}} \triangleright^{\kappa} A \qquad \Gamma \vdash_{\Delta}}{\Gamma \vdash^{\mathcal{L}}_{\Delta} \mathsf{prev}\kappa.t :_{\mathcal{I}} \forall \kappa.A}$$

Reduction rules

$$\mathsf{freeze}^{\kappa}_{l'}(\mathsf{next}^{\kappa}\, l.t) \to t\left[l'/l\right]$$

$$\mathsf{next}^{\kappa}\, l.(\mathsf{freeze}^{\kappa}_{l}\, t) \to t$$

$$\mathsf{prev}\kappa.\mathsf{next}^{\kappa}\, l.t \to \Lambda\kappa.t[\mathsf{freeze}^{\kappa}_{l}\, x \mapsto (\mathsf{prev}\kappa.x)[\kappa]]$$

Reduction rules

$$\mathsf{freeze}^{\kappa}_{l'}(\mathsf{next}^{\kappa}\, l.t) \to t\left[l'/l\right]$$

$$\mathsf{next}^{\kappa}\xi\left[x \leftarrow \mathsf{next}^{\kappa}\xi.u\right].A = \mathsf{next}^{\kappa}\xi.A\left[u/x\right]$$

$$\mathsf{next}^{\kappa}\, l.(\mathsf{freeze}^{\kappa}_{l}\, t) \to t$$

$$\mathsf{next}^{\kappa}\xi\left[x \leftarrow t\right].x = t$$

$$\mathsf{prev}\kappa.\mathsf{next}^{\kappa}\, l.t \to \Lambda\kappa.t[\mathsf{freeze}^{\kappa}_{l}\, x \mapsto (\mathsf{prev}\kappa.x)[\kappa]]$$

$$\mathsf{prev}\kappa.(\mathsf{next}^{\kappa}\xi.t) = \Lambda\kappa.t(\mathsf{adv}^{\kappa}\,(\xi))$$

$$\mathsf{adv}^{\kappa}\,([x_1 \leftarrow t_1, \ldots x_n \leftarrow t_n]) = [(\mathsf{prev}\kappa.t_1)[\kappa]/x_1, \ldots, (\mathsf{prev}\kappa.t_n)[\kappa]/x_n]$$

η-rule for $\triangleright$

$$\mathsf{next}^{\kappa}\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}^{\kappa}\,x/y] = \mathsf{next}^{\kappa}\,[x \leftarrow t]\,\xi.u\,[t/y]$$

η-rule for $\triangleright$

$$\mathsf{next}^\kappa\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}^\kappa\,x/y] = \mathsf{next}^\kappa\,[x \leftarrow t]\,\xi.u\,[t/y]$$

$$\mathsf{next}^\kappa\,l.(\mathsf{freeze}^\kappa_l\,t) \to t$$

η-rule for $\triangleright$

$$\mathsf{next}^\kappa\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}^\kappa\,x/y] = \mathsf{next}^\kappa\,[x \leftarrow t]\,\xi.u\,[t/y]$$

$$\mathsf{next}^\kappa\, l.(\mathsf{freeze}^\kappa_{l'}\, t) \to t \qquad l \notin \mathsf{fl}(t)$$

η-rule for $\triangleright$

$$\mathsf{next}^\kappa\,[x \leftarrow t]\,\xi.u\,[\mathsf{next}^\kappa\,x/y] = \mathsf{next}^\kappa\,[x \leftarrow t]\,\xi.u\,[t/y]$$

$$\mathsf{next}^\kappa\, l.(\mathsf{freeze}^\kappa_{l'}\, t) \to t \qquad l \notin \mathsf{fl}(t)$$

This rule breaks confluence!

Future Work

What we have

- confluence proof
- strong normalisation without dependent types
- completeness w.r.t. delayed substitution calculus

Future Work

What we have

- confluence proof
- strong normalisation without dependent types
- completeness w.r.t. delayed substitution calculus

What is missing

- strong normalisation of dependently typed calculus
- soundness w.r.t. delayed substitution calculus